Tables (12)

Name	Type	Schema
Delivery_Expenses		CREATE TABLE Delivery_Expenses (Expense_ID INTEGER PRIMARY KEY AUTOINCREMENT, Order_ID INTEGER, Amount REAL NOT NULL, Expense_Date DATETIME DEFAULT CURRENT_TIMESTAMP, IsDeleted INTEGER DEFAULT 0, FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID))
Expense_ID	INTEGER	"Expense_ID" INTEGER
Order_ID	INTEGER	"Order_ID" INTEGER
Amount	REAL	"Amount" REAL NOT NULL
Expense_Date	DATETIME	"Expense_Date" DATETIME DEFAULT CURRENT_TIMESTAMP
IsDeleted	INTEGER	"IsDeleted" INTEGER DEFAULT 0
Discounts		CREATE TABLE Discounts (Discount_ID INTEGER PRIMARY KEY AUTOINCREMENT, Shopkeeper_ID INTEGER, Order_ID INTEGER, Discount_Amount REAL NOT NULL, Applied_By TEXT NOT NULL, Discount_Date DATETIME DEFAULT CURRENT_TIMESTAMP, IsDeleted INTEGER DEFAULT 0, FOREIGN KEY (Shopkeeper_ID) REFERENCES Shopkeepers(ID), FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID))
Discount_ID	INTEGER	"Discount_ID" INTEGER
Shopkeeper_ID	INTEGER	"Shopkeeper_ID" INTEGER
Order_ID	INTEGER	"Order_ID" INTEGER
Discount_Amount	REAL	"Discount_Amount" REAL NOT NULL
Applied_By	TEXT	"Applied_By" TEXT NOT NULL
Discount_Date	DATETIME	"Discount_Date" DATETIME DEFAULT CURRENT_TIMESTAMP
IsDeleted	INTEGER	"IsDeleted" INTEGER DEFAULT 0
Inventory_Updates		CREATE TABLE Inventory_Updates (Update_ID INTEGER PRIMARY KEY AUTOINCREMENT, Product_SKU TEXT, Old_Quantity INTEGER NOT NULL, New_Quantity INTEGER NOT NULL, Updated_By TEXT NOT NULL, Update_Date DATETIME DEFAULT CURRENT_TIMESTAMP, IsDeleted INTEGER DEFAULT 0, FOREIGN KEY (Product_SKU) REFERENCES Products(SKU))
Update_ID	INTEGER	"Update_ID" INTEGER
Product_SKU	TEXT	"Product_SKU" TEXT
Old_Quantity	INTEGER	"Old_Quantity" INTEGER NOT NULL
New_Quantity	INTEGER	"New_Quantity" INTEGER NOT NULL
Updated_By	TEXT	"Updated_By" TEXT NOT NULL
Update_Date	DATETIME	"Update_Date" DATETIME DEFAULT CURRENT_TIMESTAMP
IsDeleted	INTEGER	"IsDeleted" INTEGER DEFAULT 0
Khata		CREATE TABLE "Khata" ("ID" INTEGER, "Shopkeeper_ID" INTEGER UNIQUE, "Brand" TEXT NOT NULL, "Total_Due" REAL DEFAULT 0, "Last_Payment_Date" DATETIME, "IsDeleted" INTEGER DEFAULT 0, UNIQUE("Shopkeeper_ID","Brand"), FOREIGN KEY("Shopkeeper_ID") REFERENCES "Shopkeepers"("ID"), PRIMARY KEY("ID" AUTOINCREMENT))
ID	INTEGER	"ID" INTEGER
Shopkeeper_ID	INTEGER	"Shopkeeper_ID" INTEGER UNIQUE
Brand	TEXT	"Brand" TEXT NOT NULL
Total_Due	REAL	"Total_Due" REAL DEFAULT 0
Last_Payment_Date	DATETIME	"Last_Payment_Date" DATETIME
IsDeleted	INTEGER	"IsDeleted" INTEGER DEFAULT 0
Order_Items		CREATE TABLE Order_Items (Order_Item_ID INTEGER PRIMARY KEY AUTOINCREMENT, Order_ID INTEGER, Product_SKU TEXT, Quantity INTEGER NOT NULL, Price REAL NOT NULL, IsDeleted INTEGER DEFAULT 0, FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID), FOREIGN KEY (Product_SKU) REFERENCES Products(SKU))
Order_Item_ID	INTEGER	"Order_Item_ID" INTEGER
Order_ID	INTEGER	"Order_ID" INTEGER

Name	Type	Schema
Product_SKU	TEXT	"Product_SKU" TEXT
Quantity	INTEGER	"Quantity" INTEGER NOT NULL
Price	REAL	"Price" REAL NOT NULL
IsDeleted	INTEGER	"IsDeleted" INTEGER DEFAULT 0
Orders		`CREATE TABLE Orders (Order_ID INTEGER PRIMARY KEY AUTOINCREMENT, Shopkeeper_ID INTEGER, Salesman_ID INTEGER, Order_Date DATETIME DEFAULT CURRENT_TIMESTAMP, Total_Amount REAL NOT NULL, IsDeleted INTEGER DEFAULT 0, Order_Info TEXT, FOREIGN KEY (Shopkeeper_ID) REFERENCES Shopkeepers(ID), FOREIGN KEY (Salesman_ID) REFERENCES Salesmen(ID))`
Order_ID	INTEGER	"Order_ID" INTEGER
Shopkeeper_ID	INTEGER	"Shopkeeper_ID" INTEGER
Salesman_ID	INTEGER	"Salesman_ID" INTEGER
Order_Date	DATETIME	"Order_Date" DATETIME DEFAULT CURRENT_TIMESTAMP
Total_Amount	REAL	"Total_Amount" REAL NOT NULL
IsDeleted	INTEGER	"IsDeleted" INTEGER DEFAULT 0
Order_Info	TEXT	"Order_Info" TEXT
Payments		`CREATE TABLE Payments (Payment_ID INTEGER PRIMARY KEY AUTOINCREMENT, Shopkeeper_ID INTEGER, Order_ID INTEGER, Salesman_ID INTEGER, Amount_Paid REAL NOT NULL, Payment_Date DATETIME DEFAULT CURRENT_TIMESTAMP, IsDeleted INTEGER DEFAULT 0, FOREIGN KEY (Shopkeeper_ID) REFERENCES Shopkeepers(ID), FOREIGN KEY (Order_ID) REFERENCES Orders(Order_ID), FOREIGN KEY (Salesman_ID) REFERENCES Salesmen(ID))`
Payment_ID	INTEGER	"Payment_ID" INTEGER
Shopkeeper_ID	INTEGER	"Shopkeeper_ID" INTEGER
Order_ID	INTEGER	"Order_ID" INTEGER
Salesman_ID	INTEGER	"Salesman_ID" INTEGER
Amount_Paid	REAL	"Amount_Paid" REAL NOT NULL
Payment_Date	DATETIME	"Payment_Date" DATETIME DEFAULT CURRENT_TIMESTAMP
IsDeleted	INTEGER	"IsDeleted" INTEGER DEFAULT 0
Products		`CREATE TABLE Products ( SKU TEXT PRIMARY KEY, Name TEXT NOT NULL, Brand TEXT NOT NULL, Size TEXT, Quantity INTEGER NOT NULL, Price REAL NOT NULL, Last_Updated DATETIME DEFAULT CURRENT_TIMESTAMP , IsDeleted INTEGER DEFAULT 0)`
SKU	TEXT	"SKU" TEXT
Name	TEXT	"Name" TEXT NOT NULL
Brand	TEXT	"Brand" TEXT NOT NULL
Size	TEXT	"Size" TEXT
Quantity	INTEGER	"Quantity" INTEGER NOT NULL
Price	REAL	"Price" REAL NOT NULL
Last_Updated	DATETIME	"Last_Updated" DATETIME DEFAULT CURRENT_TIMESTAMP
IsDeleted	INTEGER	"IsDeleted" INTEGER DEFAULT 0
Returns		`CREATE TABLE Returns (Return_ID INTEGER PRIMARY KEY AUTOINCREMENT, Order_ID INTEGER, Product_SKU TEXT, Quantity_Returned INTEGER NOT NULL, Return_Date DATETIME DEFAULT CURRENT_TIMESTAMP, Reason TEXT, IsDeleted INTEGER DEFAULT 0, FOREIGN KEY(Order_ID) REFERENCES Orders(Order_ID), FOREIGN KEY(Product_SKU) REFERENCES Products(SKU))`
Return_ID	INTEGER	"Return_ID" INTEGER
Order_ID	INTEGER	"Order_ID" INTEGER
Product_SKU	TEXT	"Product_SKU" TEXT
Quantity_Returned	INTEGER	"Quantity_Returned" INTEGER NOT NULL
Return_Date	DATETIME	"Return_Date" DATETIME DEFAULT CURRENT_TIMESTAMP

Name	Type	Schema
Reason	TEXT	"Reason" TEXT
IsDeleted	INTEGER	"IsDeleted" INTEGER DEFAULT 0
Salesmen		CREATE TABLE Salesmen (ID INTEGER PRIMARY KEY AUTOINCREMENT, Name TEXT NOT NULL, Contact_Info TEXT, Total_Sales REAL DEFAULT 0, Total_Collections REAL DEFAULT 0 , IsDeleted INTEGER DEFAULT 0)
ID	INTEGER	"ID" INTEGER
Name	TEXT	"Name" TEXT NOT NULL
Contact_Info	TEXT	"Contact_Info" TEXT
Total_Sales	REAL	"Total_Sales" REAL DEFAULT 0
Total_Collections	REAL	"Total_Collections" REAL DEFAULT 0
IsDeleted	INTEGER	"IsDeleted" INTEGER DEFAULT 0
Shopkeepers		CREATE TABLE Shopkeepers (ID INTEGER PRIMARY KEY AUTOINCREMENT, Name TEXT NOT NULL, Contact_Info TEXT , IsDeleted INTEGER DEFAULT 0, Brand TEXT NOT NULL DEFAULT 'Candyland')
ID	INTEGER	"ID" INTEGER
Name	TEXT	"Name" TEXT NOT NULL
Contact_Info	TEXT	"Contact_Info" TEXT
IsDeleted	INTEGER	"IsDeleted" INTEGER DEFAULT 0
Brand	TEXT	"Brand" TEXT NOT NULL DEFAULT 'Candyland'
sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq)
name		"name"
seq		"seq"

Indices (0)

Name	Type	Schema

Views (0)

Name	Type	Schema

Triggers (10)

Name	Type	Schema
create_khata_on_shopkeeper_insert		CREATE TRIGGER create_khata_on_shopkeeper_insert AFTER INSERT ON Shopkeepers FOR EACH ROW BEGIN INSERT INTO Khata (Shopkeeper_ID, Brand, Total_Due, Last_Payment_Date, IsDeleted) VALUES (NEW.ID, NEW.Brand, 0, DATE('now'), 0); END
reduce_product_quantity_on_insert		CREATE TRIGGER reduce_product_quantity_on_insert AFTER INSERT ON Order_Items FOR EACH ROW BEGIN UPDATE Products SET Quantity = Quantity - NEW.Quantity, Last_Updated = CURRENT_TIMESTAMP WHERE SKU = NEW.Product_SKU; END
restore_product_quantity_on_delete		CREATE TRIGGER restore_product_quantity_on_delete AFTER UPDATE OF IsDeleted ON Order_Items FOR EACH ROW WHEN NEW.IsDeleted = 1 AND OLD.IsDeleted = 0 BEGIN UPDATE Products SET Quantity = Quantity + OLD.Quantity, Last_Updated = CURRENT_TIMESTAMP WHERE SKU = OLD.Product_SKU; END
update_khata_on_order_delete		CREATE TRIGGER update_khata_on_order_delete AFTER UPDATE OF IsDeleted ON Orders FOR EACH ROW WHEN NEW.IsDeleted = 1 AND OLD.IsDeleted = 0 BEGIN UPDATE Khata SET Total_Due = (SELECT

Name	Type	Schema
		COALESCE(SUM(o.Total_Amount - (o.Total_Amount * COALESCE(d.Total_Discount, 0) / 100)), 0) FROM Orders o LEFT JOIN (SELECT Order_ID, SUM(Discount_Amount) AS Total_Discount FROM Discounts WHERE IsDeleted = 0 GROUP BY Order_ID) d ON o.Order_ID = d.Order_ID WHERE o.Shopkeeper_ID = OLD.Shopkeeper_ID AND o.IsDeleted = 0) - (SELECT COALESCE(SUM(p.Amount_Paid), 0) FROM Payments p WHERE p.Shopkeeper_ID = OLD.Shopkeeper_ID AND p.IsDeleted = 0), Last_Payment_Date = (SELECT MAX(Payment_Date) FROM Payments WHERE Shopkeeper_ID = OLD.Shopkeeper_ID AND IsDeleted = 0) WHERE Shopkeeper_ID = OLD.Shopkeeper_ID; END
update_khata_on_order_insert		CREATE TRIGGER update_khata_on_order_insert AFTER INSERT ON Orders FOR EACH ROW BEGIN UPDATE Khata SET Total_Due = (SELECT COALESCE(SUM(o.Total_Amount - (o.Total_Amount * COALESCE(d.Total_Discount, 0) / 100)), 0) FROM Orders o LEFT JOIN (SELECT Order_ID, SUM(Discount_Amount) AS Total_Discount FROM Discounts WHERE IsDeleted = 0 GROUP BY Order_ID) d ON o.Order_ID = d.Order_ID WHERE o.Shopkeeper_ID = NEW.Shopkeeper_ID AND o.IsDeleted = 0) - (SELECT COALESCE(SUM(p.Amount_Paid), 0) FROM Payments p WHERE p.Shopkeeper_ID = NEW.Shopkeeper_ID AND p.IsDeleted = 0), Last_Payment_Date = (SELECT MAX(Payment_Date) FROM Payments WHERE Shopkeeper_ID = NEW.Shopkeeper_ID AND IsDeleted = 0) WHERE Shopkeeper_ID = NEW.Shopkeeper_ID; END
update_khata_on_order_update		CREATE TRIGGER update_khata_on_order_update AFTER UPDATE ON Orders FOR EACH ROW BEGIN UPDATE Khata SET Total_Due = (SELECT COALESCE(SUM(o.Total_Amount - (o.Total_Amount * COALESCE(d.Total_Discount, 0) / 100)), 0) FROM Orders o LEFT JOIN (SELECT Order_ID, SUM(Discount_Amount) AS Total_Discount FROM Discounts WHERE IsDeleted = 0 GROUP BY Order_ID) d ON o.Order_ID = d.Order_ID WHERE o.Shopkeeper_ID = NEW.Shopkeeper_ID AND o.IsDeleted = 0) - (SELECT COALESCE(SUM(p.Amount_Paid), 0) FROM Payments p WHERE p.Shopkeeper_ID = NEW.Shopkeeper_ID AND p.IsDeleted = 0), Last_Payment_Date = (SELECT MAX(Payment_Date) FROM Payments WHERE Shopkeeper_ID = NEW.Shopkeeper_ID AND IsDeleted = 0) WHERE Shopkeeper_ID = NEW.Shopkeeper_ID; END
update_khata_on_payment_delete		CREATE TRIGGER update_khata_on_payment_delete AFTER UPDATE OF IsDeleted ON Payments FOR EACH ROW WHEN NEW.IsDeleted = 1 AND OLD.IsDeleted = 0 -- Only trigger when marking as deleted BEGIN UPDATE Khata SET Total_Due = Total_Due + OLD.Amount_Paid WHERE Shopkeeper_ID = OLD.Shopkeeper_ID AND Brand = (SELECT Brand FROM Shopkeepers WHERE ID = OLD.Shopkeeper_ID); END
update_khata_on_payment_insert		CREATE TRIGGER update_khata_on_payment_insert AFTER INSERT ON Payments FOR EACH ROW BEGIN -- Update the total due by subtracting the payment amount UPDATE Khata SET Total_Due = Total_Due - NEW.Amount_Paid, Last_Payment_Date = NEW.Payment_Date WHERE Shopkeeper_ID =

Name	Type	Schema
		NEW.Shopkeeper_ID AND Brand = (SELECT Brand FROM Shopkeepers WHERE ID = NEW.Shopkeeper_ID); END
update_khata_on_payment_update		CREATE TRIGGER update_khata_on_payment_update AFTER UPDATE ON Payments FOR EACH ROW BEGIN -- Add back old payment amount before subtracting new payment amount UPDATE Khata SET Total_Due = Total_Due + OLD.Amount_Paid - NEW.Amount_Paid, Last_Payment_Date = NEW.Payment_Date WHERE Shopkeeper_ID = NEW.Shopkeeper_ID AND Brand = (SELECT Brand FROM Shopkeepers WHERE ID = NEW.Shopkeeper_ID); END
update_product_quantity_on_update		CREATE TRIGGER update_product_quantity_on_update AFTER UPDATE OF Quantity ON Order_Items FOR EACH ROW BEGIN UPDATE Products SET Quantity = Quantity - (NEW.Quantity - OLD.Quantity), Last_Updated = CURRENT_TIMESTAMP WHERE SKU = NEW.Product_SKU; END